The Reasonably unreasonable effectiveness of RNNs

Venchislaw

National University of City17

Newbie Deep Learning Engineer dramatically applies Feynman's Technique!!!

How dare he??

venchislavcodes@gmail.com

Abstract

Our brain processes information sequentially, keeping in mind context.

For example, you are not processing each word in vacuum while reading this text.

However, ordinary Neural Networks (like MLP) can not do it.

We can process each word individually, but not sequentially.

It may seem okay until the following tasks:

1) Detect whether “Teddy” is a person’s name in text.

   Easy, right?

   “USA former president Teddy Roosevelt”

   “Plush toy Teddy Bear on the bed”

2) Predict the next word in text.

There appears to be a lot of tasks to be processed sequentially.

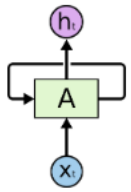
Sequence processing becomes a separate problem, which can be tackled by RNNs!

Keywords: RNN, Sequence, LSTM/GRU

Intro

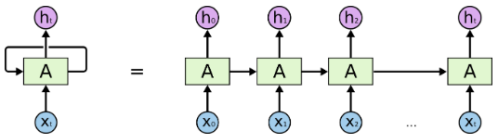
What?

RNN (recurrent neural network) - neural network architecture with inner loops to hold previous sequence context.



Recurrent Neural Networks have loops.

Personally I find these schemes confusing, as they do not depict the entire picture.



An unrolled recurrent neural network.

RNN inside.

Each RNN block takes sequence element as input (it can be character/word/sound) and produces two values:

$a^{<t>}$ - hidden state for current t_{th} element of a sequence. Also denoted as $h^{<t>}$

$y^{<t>}$ - output for the current sequence block. It can be nothing, if we don't want to make output for the current element of a sequence, or boolean True/False if we want to classify, whether each word is a name or not.

$a^{<t>}$ flows to the $t+1$ block as an additional context input.

$a^{<0>}$ is usually set to be vector of zeros $\vec{0}$

This is really nice, but…

RNNs are not good enough. Yes, they hold some context from the past & process sequence properly, but there are few drawbacks of plain RNN:

1) Includes context from the past only

   "Teddy Roosevelt" and "Teddy Bear" begin the same. To find any difference we need to process data from "right-to-left". I think it's something human-like, where we read the word, extend its meaning with later context and have this "aha moment".

2) Loses context through the time.

   "I'm from France. I like to eat croissants, baguettes, … (blah blah blah). My Native Language is French"

   Despite the distance between these two words they are connected to each other. However plain RNN can easily Lose context and say "Native Language is German"

Keep It mind.

Let's start with the solution to the 2nd problem:

Why does our network lose context?

- Gradient Vanishing

It would be nice, if our RNN would have some sort of "Long-Term memory" to keep in mind important information (native country of a subject/its gender/is it plural).

Who said we can't do this?

LSTM (long short-term memory) comes to the rescue!

Concept is pretty simple.

Key idea is to keep long-term memory across the sequence, adding new information and forgetting irrelevant information.

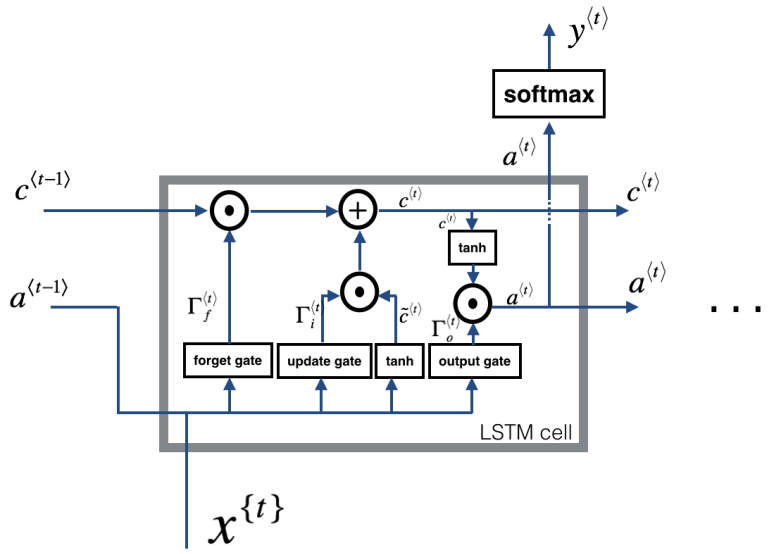
Why Forgetting?

We need to forget irrelevant information. For example, when the subject of speech changes, we need to forget the characteristics of the previous subject and start keeping track of the new subject's characteristics.

How do we do it?

All these methods are implemented with "gates".

Let's see the entire picture first:



$$\Gamma_f^{\langle t \rangle} = \sigma(W_f[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_f)$$
$$\Gamma_u^{\langle t \rangle} = \sigma(W_u[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_u)$$
$$\tilde{c}^{\langle t \rangle} = \tanh(W_C[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_C)$$
$$c^{\langle t \rangle} = \Gamma_f^{\langle t \rangle} \circ c^{\langle t-1 \rangle} + \Gamma_u^{\langle t \rangle} \circ \tilde{c}^{\langle t \rangle}$$
$$\Gamma_o^{\langle t \rangle} = \sigma(W_o[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_o)$$
$$a^{\langle t \rangle} = \Gamma_o^{\langle t \rangle} \circ \tanh(c^{\langle t \rangle})$$

Forget gate:

Just a sigmoid function applied to the weighted sum of input and previous hidden state.

Update Gate:

Updates our long-term memory with new info.

It calculates “candidate” $\tilde{c}$ to add to our Long-term memory.

It uses new information and short-term memory.

Update Gate consists of 2 parts:

1) Calculate Percentage % of new information to add (uses sigmoid)
2) Calculate “candidate” via Tanh.

In some sources Update Gate is also referred as “Input Gate”

Output Gate:

It squishes our long-term memory via Tanh to the range [-1; 1] and determines the percentage of current short-term memory to use via sigmoid. It produces new short-term memory value.

We can also modify it and pass it through some activation function to get output for the current sequence block (if needed).

Conclusion for LSTM.

LSTM (long short-term memory) resolves an issue of Vanishing gradient in sequence models.

I find name of it interesting because it unwraps two important ideas:

1) We have both long and short-term memories separated
2) Actually long-term memory is just a short-term memory carried through the sequence with small changes.

GRU (not bald one)

Jokes aside, GRU (gated recurrent unit) is an “optimized” version of LSTM with less computations, but, as a result, worse performance (not in general. In some tasks GRUs perform better).

It avoids separation of Long-Term memory and Short-term memory.

GRU uses one hidden state variable (let’s call it $c^{<t>}$ for convenience).

Avoiding this separation reduces computational complexity of the block.

As a result we don’t have an “output gate”.

What happens:

1) Forget Gate.

   Similarly to LSTM we forget some irrelevant information

2) Update Gate.

   Updates hidden state memory $c^{<t>}$ with new information.

Update Rule For GRU:

$$\Gamma_r = \sigma(Wr[c^{<t-1>}, x^{<t>}])$$

$$\tilde{c}^{<t>} = tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

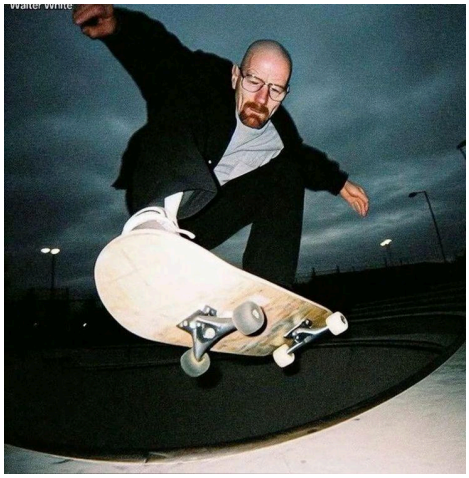
(sorry for the quality of the last equation. I had to render it directly with quicklatex.com)

References

Lastname, C. (2008). Title of the source without caps except Proper Nouns or: First word after colon. *The Journal or Publication Italicized and Capped*, Vol#(Issue#), Page numbers.

Lastname, O. (2010). Online journal using DOI or digital object identifier. *Main Online Journal Name*, Vol#(Issue#), 159-192. doi: 10.1000/182

Lastname, W. (2009). If there is no DOI use the URL of the main website referenced. *Article Without DOI Reference*, Vol#(Issue#), 166-212. Retrieved from http://www.example.com



Kto Pizdel?